IPMI v0.9 to IPMI v1.0
Change Summary and
Porting Considerations

Document Revision 2

October 1, 1998

Table of Contents

1. Introduction

This paper presents an overview of the changes between IPMI v0.9 and v1.0. This is not intended to be an absolute listing of all changes, but to serve as a starting point for understanding the tasks that may be involved in porting an IPMI v0.9 design to create an IPMI v1.0 system.

Towards that end, this paper also includes sections that present porting considerations by area, such as BIOS, Drivers, and Firmware. At the end of this document are tables that provide a high-level comparison of the v1.0 commands to the v0.9 changes.

2. IPMI v1.0 Goals

The first step in understanding the changes between IPMI v0.9 and IPMI v1.0 is to review the goals that guided the creation of the v1.0 specification.

- Incorporate learnings from the v0.9 "early adopter" specification to create a supportable, robust specification suitable for industry adoption. This includes adding explanatory text and examples to eliminate ambiguities.
- Enhance basic functions of the specification, (such as Deassertion Events, Entity Association Records, or Block Transfer Interface).
- Simplify structures and commands, where feasible, to either:
  - significantly reduce coding required for implementation
  - significantly reduce confusion or misinterpretation of the specification
  - improve performance
  - eliminate unused fields and commands from the specification
- Provide additional standardized functions to increase base value over v0.9, such as::
  - Watchdog Timer
  - Power Control / Reset commands
  - FRU Records for Power Supply
  - FRU Records for Management Access Address
- Don't change commands or interfaces unless the change is required to satisfy one or more of above goals. Attempt to add new functionality without changing existing commands or fields.

3. Interesting Numbers

While the effort to cover a change is not necessarily proportional to the percentage of change, it may be interesting to examine the percentage of changed commands between v0.9 and v1.0. The majority of the IPMI v1.0 specification was carried over from v0.9. Using commands as an example, here's a tally of the changed commands in IPMI v1.0:

IPMI v1.0 includes 66 total commands

- 7 may be considered as "significantly changed" from v0.9 (additional fields, plus bit fields moved)
- 5 commands with additional fields, generally static
- 4 with bits moved
- 2 with just the version number changed

- 48 commands unchanged, new and optional, or dropped:
  - 36 unchanged, or with new options bits (bits that were reserved in v0.9 but are new options in v1.0)
  - 10 new
  - 2 dropped from the specification

About 82% of the commands are unchanged or have minor changes. Of the changed commands, the majority of the changes included an extension of the existing v0.9 functionality.

4. Why Not More Backward-Compatibility?

The IPMI v1.0 focus was on making a specification suitable for broad industry adoption. Version 0.9 of the specification was created as a vehicle for industry review. In parallel, v0.9 implementations were deployed to test and validate the specification. Thus, the point of v0.9 was learn what areas needed to be cleaned up or extended for v1.0. The point of v1.0 was to incorporate those learnings.

When faced with retaining inconsistent, limiting, confusing, or hard-to-use structures in the specification for the sake of maintaining backward compatibility, the general consensus was to "clean up" those areas of the specification so that future versions of the specification would not be burdened with a legacy of those "eccentricities". As a result, there is an interim effort for early adopters to move to the v1.0 base.

Since v1.0 is considered to be a clean and solid foundation, backward compatibility with v1.0 will be a major consideration in any specification changes or future versions of IPMI.

5. Mixed Systems

In general, it is recommended that, if possible, all elements be moved to v1.0 at the same time. That is, migrate to v1.0 and avoid mixing v1.0 and v0.9 versions of controllers and software.

If this is not feasible, then the primary recommendation is to modify your system management software so that it can work with either IPMI v1.0 or v0.9 systems—but have all controllers in a system use the same IPMI version. Do not mix v1.0 and v0.9 controllers the same system.

A significantly greater effort, but technically feasible approach, would be to modify your BMC and system management software so that it can work with a v1.0 Baseboard Management Controller (BMC), and v0.9 or v1.0 satellite controllers. This requires modifying the BMC firmware so that the init agent can initialize either v1.0 or v0.9 controllers. Both v1.0 and v0.9 controllers would be described using v1.0 Sensor Data Records (SDRs). System management software would need to be able to format commands for the controllers based on their version (obtained via the Get Device ID command). Creating a BMC and software that supports a mixed system would generally be done to avoid porting v0.9 satellite controllers to v1.0. In most cases, however, the effort required for the BMC and software changes will outweigh the effort to port the satellite controller. This approach would also add significant complexity to your validation efforts, and likely prolong the time it takes to get systems over to v1.0.

6. Change Summary

The following presents a brief overview of the changes between v0.9 and v1.0. Refer to the *New for IPMI v1.0* section of the *Intelligent Platform Management Interface v1.0 Specification* for additional information.

6.1 Sensor Command Changes

The sensor commands have been changed to support:

- Deassertion Events and status.
- Consistency between thresholds and event status bits between commands and between the commands and the threshold and mask fields in the SDRs.
- New optional *Set Sensor Type* and *Get Sensor Type* commands to enable generic management controllers that do not require pre-defined types associated with their discrete and analog/threshold-based sensors.

6.2 SDR Changes

- Entity IDs have been extended and an Entity Instance field defined.
- New Device Locator SDRs. The prior Device Locator record has been split into three:
  - Management Controller Device Locator
  - FRU Device Locator
  - Generic Device Locator
- New "BMC Message Channel Info" SDR. Reports the available message channels and their types.
- New "Management Controller Confirmation" SDR (optional).
- Thresholds and Mask Bits are consistent with the sensor commands.
- Support for "modal" SDR Repository (allows implementation where repository is only updated when in an "SDR Repository Update Mode"). New commands and a completion code added to support this: *Enter/Exit SDR Update Mode*.

6.3 Messaging Interface and Messaging Command Changes

- Created "message channels" that eliminate the need for separate "buffer" commands such as *Get SMS Message Buffer* (except for the *Event Message Buffer*, which remains).
- Changes were made to the *Get Interrupt Flags* and *Clear Interrupt Flags* commands. These were renamed to *Get Message Flags* and *Clear Message Flags*, respectively.
- Changes to the *Get/Set BMC Global Enables* commands.
- The *Master Write I^2C* command has been replaced by the *Send Message* command as the way for sending IPMI messages to other interfaces and controllers. The *Get SMS Message Buffer* command has been replaced by the *Get Message* command. This was done to simplify understanding the messaging interfaces and to support future out-of-band communication extensions.
- The *Master Read I^2C* and *Master Write I^2C* were removed. Since the *Master Write-Read I^2C* command can handle both functions, the other commands were redundant.
- A new *Enable Message Channel Receive* command has been defined to allow message channels to be individually enabled/disabled.

- Explicit interfaces for System Management Interrupt (SMI) Handlers have been removed. The implementation and operation of SMI interfaces and SMI Handlers are very platform-specific. In addition, special coordination may be required between SMI Handlers and the Operating System. Thus, instead of attempting to extend the specification to fully specify SMI operations, explicit support for communications with an SMI Handler has been removed from the specification. Note that there are sufficient OEM bits and flags in the messaging commands and communication interfaces to allow an implementation to provide proprietary support for an SMI Handler.

6.4 Event Message Changes

An IPMI v1.0 BMC can accept v1.0 and v0.9 Event Messages for logging to the SEL. Version information in the SEL Event Record allows system software to differentiate v1.0 Events from v0.9 Events. The Event Record format for v1.0 Event Messages has been changed as follows:

- Event Dir bit: There's a new bit defined for Event Messages to indicate whether an event was on an assertion or the deassertion of a state. This necessitated a shift in the Event/Reading Type codes.
- Event Data 1 change: The Event Data 1 byte was changed to clearly delineate whether the Event Data 2 and/or Event Data 3 bytes hold OEM or unspecified data.
- Version Number: The EvM Revision field is set to 01h to indicate the format change.
- In addition, there are some new events that have been defined. (Refer to the Sensor Type Codes table in the specification.)

6.5 FRU Changes

The Read/Write FRU Inventory commands have been changed to provide simpler support for multiple FRU devices behind a management controller. Additional FRU information record types have been defined, but, with the exception of an update of the format version number, v1.0 FRU is backward compatible with v0.9.

- A "FRU Device ID" parameter has been added to the *Read/Write FRU* commands to allow more than four FRU devices to be accessed behind a given management controller.
- There are new, optional, multirecord types for providing alternate access address information for system management (see the FRU specification).
- Power supply and power consumption FRU records have been incorporated into the FRU specification.
- The FRU format version number has been updated to 02h from 01h.

6.6 Miscellaneous

See the following tables for other command additions and changes:

- *Get Device ID* command: added manufacturer ID information.
- *Cold Reset/Warm Reset* commands: made optional.
- New optional commands for setting and reporting ACPI power state.
- A new optional command for returning a device GUID.
- A new *Run Initialization Agent* command to support modal SDR repository.
- The *Get BT Interface Capabilities* command, added to support BT interfaces.
- A new, optional, *POH Counter* command has been defined to support a standardized interface to a power-on hours counter.

7. Change Analysis by Area

7.1 BIOS

There are no mandatory BIOS interactions specified for v0.9 or v1.0 implementations. Thus, the amount of changes to BIOS are dependent on how much optional interaction a v0.9 system previously had with BIOS. The following presents some areas that have changed that would affect BIOS interactions with IPMI.

- Changes have been made to the commands used to talk to other controllers behind the BMC:
  - The *Write I^2C* and *Get SMS Message Buffer* commands have been essentially renamed as *Send Message* and *Get Message* commands. The new commands have an additional parameter. This parameter is just a constant, fixed byte as far as BIOS communications with the BMC and other management controllers is concerned.
  - The *Get/Set Interrupt Flags* (renamed to Get/Set Message Flags) have had some bits moved around - so there may need to be a change to any routines that BIOS uses to enable interrupts from the BMC.
- SMI interrupt enabling: The *Get/Set BMC Global Enables,* and *Get/Set Interrupt Flags* (renamed *to Get/Set Message Flags*) have had some bits moved around, and explicit support for SMI has been removed. However, OEM status and enable bits are available to allow a BMC implementation to support an SMI Handler.
- Event Messages. If BIOS was logging events to the SEL, the changes in Event Message mean that the constant values that BIOS sends in events need to be adjusted to new constant values.
  - There's been a new bit defined for Event Messages to indicate whether an event was on an assertion or the deassertion of a state. This necessitated a shift in the Event/Reading Type codes.
  - The *Event Data 1* byte was changed to clearly delineate whether the Event Data 2 and/or Event Data 3 bytes hold OEM or unspecified data.
  - The EvM Revision field is set to 01h to indicate the format change.
- The *Get Device ID* command changed. Some systems may have had BIOS use this command to verify the presence of satellite management controllers. For these systems, BIOS may need to adjust to the fact that the v1.0 version returns additional parameters, and returns a version number of 1.0 for the command set info from a controller.
- The *Master Write I^2C* and *Master Read I^2C* commands have been removed from the specification. The *Master Write-Read I^2C* command replaces both. BIOS may have used these commands for accessing non-intelligent I^2C devices or satellite controllers via the BMC. In this case, routines may need to be modified to use the *Master Write-Read I^2C command* in place of the *Master Write I^2C* or *Master Read I^2C* command.

7.2 Utilities

- SDR Creation and SDR Viewing Utilities will require moderate to significant updating. Effort will vary based on the modularity of your initial code structure. The general task of porting requires altering routines that parse and assemble SDRs.
  - The majority of the content of the SDRs is unchanged—but new fields have been added and some existing fields have moved. Thus, parsing and assembly routines will need to be modified.
  - The Device Locator record has been split into three records, Management Controller Device Locator Record, FRU Device Locator Record, and Generic Device Locator Record. This was done to make SDRs easier to implement and understand. The Generic Device Locator record is for non-FRU, non-management controller devices on the IPMB or private busses. Most

implementations avoid these type of devices, since they are not supportable with generic, cross-platform software. Therefore, support for this record may be able to be phased in when needed.

- There's a new "Entity-Association" record that is used for identifying the relationship between entities to system management software. You may want to extend your SDR Creation utility to handle this new record, possibly provide additional code to make it easier to enter and view the resulting relationships between Entities.
- Management Controller Confirmation record. This is a new record that can be used to record that a given controller has been discovered in the system. You may want to extend your SDR Creation and Viewing utilities to handle this new record. You could also use this record as part of creating an automated process that discovers satellite management controllers, extracts Device SDRs from them, and updates the SDR Repository accordingly.

- FRU Update utilities require minor updating.
  - There has been a FRU Device ID parameter added to the *Read/Write FRU* commands. This was done to make support of multiple FRU devices behind a single management controller more straightforward.
  - There are new multirecord types, but the record formatting follows the present v0.9 specification. A utility may need to be extended to support the new record types.
- The SEL Viewing Utility requires minor-to-moderate updating.
  - Some new event types have been defined.
  - There's been a new bit defined for Event Messages to indicate whether an event was on an assertion or the deassertion of a state. This necessitated a shift in the Event/Reading Type codes.
  - The Event Data 1 byte was changed to clearly delineate whether the Event Data 2 and/or Event Data 3 bytes hold OEM or unspecified data.
  - The EvM Revision field is set to 01h to indicate the format change.
- Firmware Update Utility. Some vendors may have provided proprietary commands that allow their management controller firmware to be field updated. The *Get Device ID* command is commonly used to verify communications with a management controller. This command has had some fields added, and also has a new version number. Firmware Update utilities that use this command may need to adjust for this.

7.3 Instrumentation Software

Instrumentation software will need modification mainly to support the changes to the Sensor Command and Sensor Data Records. For the most part, these changes require accessing different fields for the same functions. If you're just looking to port existing functionality to v1.0, then most of the changes will involve changing constants and adapting the software to access different bits. Handling new capabilities, such as deassertion event status, will require code extensions.

- Sensor Command Changes. The main impact is in the sensor commands.
  - Bits have been added to support deassertion events.
  - The sensor commands have been changed so that the bit positions are consistent. For new implementations, this should simplify software. For porting v0.9, this means accessing different bit positions for the same functions.
  - Digital and Discrete sensor commands have merged so there are no longer special formats used the parameters for digital sensors ("digital" sensors remain as a concept, but they are implemented as a two-state discrete sensor). For new implementations, this should result in simpler software. For v0.9 routines, you can continue to parse "digital" and discrete separately if it fits your code

structure better (since the Event/Reading Type code still allows you to tell which sensors are "digital" [two state] and which are discrete [more than two states]).

- SDR Changes.
  - The Device Locator record has been split into three records, Management Controller Device Locator Record, FRU Device Locator Record, and Generic Device Locator Record. This was done to make SDRs easier to implement and understand. The Generic Device Locator record is for non-FRU, non-management controller devices on the IPMB or private busses. This is mainly for OEM support. The impact on software will be based on the structure of the code you used for parsing v0.9 Device Locator Records. In many cases, the simplest porting approach will be to modify the existing routine to accept all three v1.0 Device Locator record types. An alternate approach would be to implement separate parsing routines for each record type.
  - Fields have moved in the SDRs. The majority of the field contents have been preserved but may have moved.
  - The "Device Location" and "Device Confirmation" fields were eliminated. This was done to support the addition of an Entity Instance field, and review input that the device confirmation aspect was not widely used. If you were using device confirmation to verify the existence of a management controller, a new "Management Controller Confirmation" SDR has been defined for v1.0
  - The Entity ID and Device Location fields were merged. The Device Location codes overlapped the Entity ID Codes so these were combined for v1.0. In addition, an Entity Instance field was defined to cleanly support multiple instances of an Entity. (In v0.9, the instance and Entity ID were "overlayed" in the one-byte Entity ID. For example, the code Processor 1 was 18h and Processor 2 was 19h. This was very restrictive, since it limited both the number of different types of entities and how many you could have of each type.) Code that uses the Entity ID or Device Location values needs to be modified to interpret the Entity ID / Entity Instance field combination instead.
  - Entity Association record. This is a new record that allows more sophisticated relationships between the Entity a sensor is associated with, and other Entities in the system. For example, the Entity Association record allows you to indicate that a sensor is associated with a pair of processors. The v0.9 mechanism of using an Entity ID to locate the SDR for the FRU associated with the Entity still exists in v1.0. If your software is just used for particular systems (it is not intended to be generic, cross-platform software) then you may elect to postpone extending it for Entity Association SDR support.
  - Management Controller Confirmation record. This is a new record that can be used to record that a given controller has been discovered in the system. Later, the record information can be used to confirm that the same management controller is still present. Entry and use of this record is optional. It is provided to support using "discovery" and Device SDRs as a mechanism for populating the SDR repository. Since this is a new, optional capability, it's your decision whether to use it.
- Event Record Changes. A shift in the Event/Reading type code, version numbers change to "1.0", and a new "Event Dir" bit that was added to identify deassertion events. The main impact to routines that interpret events will be handling the new Event/Reading Type code values.
  - Some new event types have been defined.
  - There's been a new bit defined for Event Messages to indicate whether an event was on an assertion or the deassertion of a state. This necessitated a shift in the Event/Reading Type codes.
  - The Event Data 1 byte was changed to clearly delineate whether the Event Data 2 and/or Event Data 3 bytes hold OEM or unspecified data.
  - The EvM Revision field is set to 01h to indicate the format change.

7.4 Driver

There should only be minor-to-moderate updates required for system interface drivers—depending on the level of support you want to provide for new capabilities.

- Your drive must change from using *Write I^2C* or *Master Write/Read I^2C* command to using the *Send Message* command for sending IPMI messages to satellite controllers on the IPMB. Similarly, instead of using the *Get SMS Message Buffer* command, you need to use the *Get Message* command to retrieve messages sent to SMS from the IPMB. These commands have an additional "channel" parameter that is a fixed constant for regular SMS-to-IPMB messaging, otherwise Message formats remain the same. (In general, your driver shouldn't parse message data, but should leave that to higher software. If your driver needs to parse messages, note that there is a new SDR, "BMC Message Channel Info" that would be used to tell you the format of messages received from each channel.)
- There's a new "Block Transfer Interface." Support for this is dependent on whether/when you want to support the BT interface.
- Flags: The *Get Interrupt Flags* command has been changed to *Get Message Flags.* Some bits have moved. The command finds when the *Receive Message Queue* has data. For v0.9, you got equivalent flags for the *SMS Message Buffer.*
- Interrupt option: the IPMI system interfaces continue to work in polled fashion. There's been provision for interrupts added to the interfaces. It's optional if/when you decide to support interrupt-driven versions of the interfaces.

7.5 SMI Handler

SMI Handler communications with the BMC is considered to be "platform specific" for IPMI v1.0. However, sufficient "OEM" flags have been provided to allow SMI Handler communication to continue to be supported.

- Some flag bits have moved in the *Get Interrupt Flags (now called "Get Message Flags")* command. The SMM related flags have been renamed "OEM flags" in the command and the system interfaces, but for our application would remain used for the SMM Message Buffer flags.
- Event Messages have had some format changes, so some constants used in SMI handler events will need to change.
    - There's been a new bit defined for Event Messages to indicate whether an event was on an assertion or the deassertion of a state. This necessitated a shift in the Event/Reading Type codes.
    - The Event Data 1 byte was changed to clearly delineate whether the Event Data 2 and/or Event Data 3 bytes hold OEM or unspecified data.
    - The EvM Revision field is set to 01h to indicate the format change

7.6 System Management Software Applications

This varies based on how abstracted your instrumentation software is. If your system management applications are based on a standard schema, such as CIM or DMI, and your instrumentation software is handling the task of populating the schema, it's possible that there would be little or no impact to your management applications.

7.7 Firmware

The biggest changes are in the sensor commands. Refer to the previous Change Summary section, and following Command Comparison table.

7.8 Evaluation

In most environments, any changes to interfaces require changes in evaluation. The degree of change will be dependent on how automated your existing v0.9 evaluation may have been. If you were mainly testing using "manual" tools and having a test engineer enter individual commands and view results, then the impact may be mainly writing a new test plan. If you have extensive automated or scripted tools, then you'll probably need to modify them.

- Changes to Messaging and FRU commands require some updates to test tools that send and receive messages from management controllers, or load/access FRU information.
- Test plan / script changes to cover:
  - New SDR types
  - Event Message format change
  - New sensor commands
  - Other command changes

7.9 Manufacturing Impact:

- Changes to messaging and FRU commands require some updates to system integration tools.
- Test tool changes to cover:
  - Event Message format change
  - New sensor commands

8. Command Comparison

The following tables compare the present IPMI v1.0 commands against the IPMI v0.9 commands.

IPM "Global" Commands

Command	v0.9, v1.0	Code
Get Device ID	v0.9	01h
	v1.0: added fields for manufacturer ID and system	01h
Cold Reset	same format v0.9 mandatory v1.0 optional	02h
Warm Reset	same format v0.9 mandatory v1.0 optional	03h
Get Self Test Results	same	04h
Manufacturing Test On	same	05h
Set ACPI Power State	v1.0: NEW, optional	06h
Get ACPI Power State	v1.0: NEW, optional	07h
Get Device GUID	v1.0: NEW, optional	08h
Broadcast Get Device ID	v0.9	01h
	v1.0: same changes as for Get Device ID, above	01h

Sensor/Event Device Commands

Command	v0.9, v1.0	Code	NetFn
Set Event Receiver	same	00h	S/E
Get Event Receiver	same	01h	S/E
Platform Event Message	v0.9	02h	S/E
Platform Event Message	v1.0: EvMRev 02h→03h, Added Event Dir bit.	02h	S/E
Get Device SDR Info	same	20h	S/E
Get Device SDR	same	21h	S/E
Get Sensor Reading Factors	same	23h	S/E
Set Sensor Hysterisis	same	24h	S/E
Get Sensor Hysterisis	same	25h	S/E
Set Sensor Threshold	v0.9	26h	S/E
	v1.0: changes for consistency	26h	S/E
Get Sensor Threshold	v0.9	27h	S/E
	v1.0: changes for consistency	27h	S/E
Set Sensor Event Message Enable	v0.9	28h	S/E
	v1.0: changes for consistency & deassertion events	28h	S/E
Get Sensor Event Message Enable	v0.9	29h	S/E
	v1.0: changes for consistency & deassertion events	29h	S/E
Re-arm Sensor Events	v0.9	2Ah	S/E
	v1.0: changes for consistency & deassertion events	2Ah	S/E
Get Sensor Event Status	v0.9	2Bh	S/E
	v1.0: changes for consistency & deassertion events, ≥ & ≤ thresholds	2Bh	S/E
Get Sensor Reading	v0.9	2Dh	S/E
	v1.0: changes for consistency, update in progress bit, ≥ & ≤ thresholds	2Dh	S/E
Set Sensor Type	v1.0: NEW optional	2Eh	S/E
Get Sensor Type	v1.0: NEW optional	2Fh	S/E

Chassis Device Commands

Command	v0.9, v1.0	Code	Net Fn
Get Chassis Status	v1.0: NEW optional	01h	Chassis
Chassis Control	v1.0: NEW optional	02h	Chassis
Get POH Counter	v1.0: NEW optional	0Fh	Chassis

FRU Inventory Device Commands

Command	v0.9, v1.0	Code	Net Fn
reserved	same	00h-0Fh	Storage
Get FRU Inventory Area Info	same	10h	Storage
Read FRU Inventory Data	v0.9	11h	Storage
	v1.0: added FRU Device ID field	11h	Storage
Write FRU Inventory Data	v0.9	12h	Storage
	v1.0: added FRU Device ID field	12h	Storage
reserved	same	13h-1Fh	Storage

SDR Repository Device Commands

Command	v0.9, v1.0	Code	Net Fn
Get SDR Repository Info	v0.9	20h	storage
	v1.0: version number change from 90h to 01h	20h	storage
Get SDR Repository Allocation Info	same	21h	storage
Reserve SDR Repository	same	22h	storage
Get SDR	same	23h	storage
Add SDR	same	24h	storage
Partial Add SDR	same	25h	storage
Delete SDR	same	26h	storage
Clear SDR Repository	same	27h	storage
Get SDR Repository Time	same	28h	storage
Set SDR Repository Time	same	29h	storage
Enter SDR Repository Update Mode	v1.0: NEW	2Ah	storage
Exit SDR Repository Update Mode	v1.0: NEW	2Bh	storage
Run Initialization Agent	v1.0: NEW	2Ch	storage
reserved	v0.9	2Ah-3Fh	storage
reserved	v1.0	2Dh-3Fh	storage

SEL Device Commands

Command	v0.9, v1.0	Code	Net Fn
Get SEL Info	v0.9	40h	storage
	v1.0: version number change from 90h to 01h	40h	storage
Get SEL Allocation Info	same	41h	storage
Reserve SEL	same	42h	storage
Get SEL Entry	v0.9	43h	storage
	v1.0: documents that this command returns all 16-bytes of SEL record	43h	storage
Add SEL Entry**	same	44h	storage
Partial Add SEL Entry	same	45h	storage
Delete SEL Entry	same	46h	storage
Clear SEL	same	47h	storage
Get SEL Time	same	48h	storage
Set SEL Time	same	49h	storage
reserved	same	4Ah-FFh	storage

BMC Application Commands

Command	v0.9, v1.0	Code	Net Fn
Set BMC Global Enables	v0.9	1Ch	App
	v1.0: bit moves, replaced SMI and SMM buffer bits with OEM bits, added receive msg queue interrupt	2Eh	
Get BMC Global Enables	v0.9	1Dh	App
	v1.0: corresponds to changes in v1.0 Set BMC Global Enables	2Fh	
Reset Watchdog Timer	same	22h	App
Set Watchdog Timer	v0.9	24h	App
	v1.0 per NS, plus new NMI timeout action	24h	
Get Watchdog Timer	v0.9	25h	App
	v1.0 per NS, plus new NMI timeout action	25h	
Clear Interrupt Flags	v0.9	30h	App
Clear Message Flags	v1.0: renamed, corresponds to changes in BMC Global Enables	30h	

Command	v0.9, v1.0	Code	Net Fn
Get Interrupt Flags	v0.9	31h	App
Get Message Flags	v1.0: renamed, corresponds to changes in BMC Global Enables	31h	App
Send Message	v1.0: NEW	34h	App
Read Event Msg Buffer	v0.9	35h	App
	v1.0: added a generic completion code: 80h = data not available (queue / buffer empty), clarified returns same 16-bytes as SEL Record	35h	App
Read SMM Msg Buffer	v0.9	36h	App
	v1.0: REMOVED		App
Read SMS Msg Buffer	v0.9	37h	App
Enable Message Channel Receive	v1.0: NEW		App
Get Message	v1.0: renamed, changed to include channel number field to support future interfaces.		App
Get BT Interface Capabilities	v1.0: NEW		App
Master Write I^2C	v0.9	50h	App
	v1.0: REMOVED		
Master Read I^2C	v0.9	51h	App
	v1.0: REMOVED		
Master Write-Read I^2C	same	52h	App
Error Report	same	FFh	App
	v1.0 REMOVED		

LAST PAGE